

Date=17/11/2020

Lecture By=Manish Mahant

Subject ⇒ States And Props

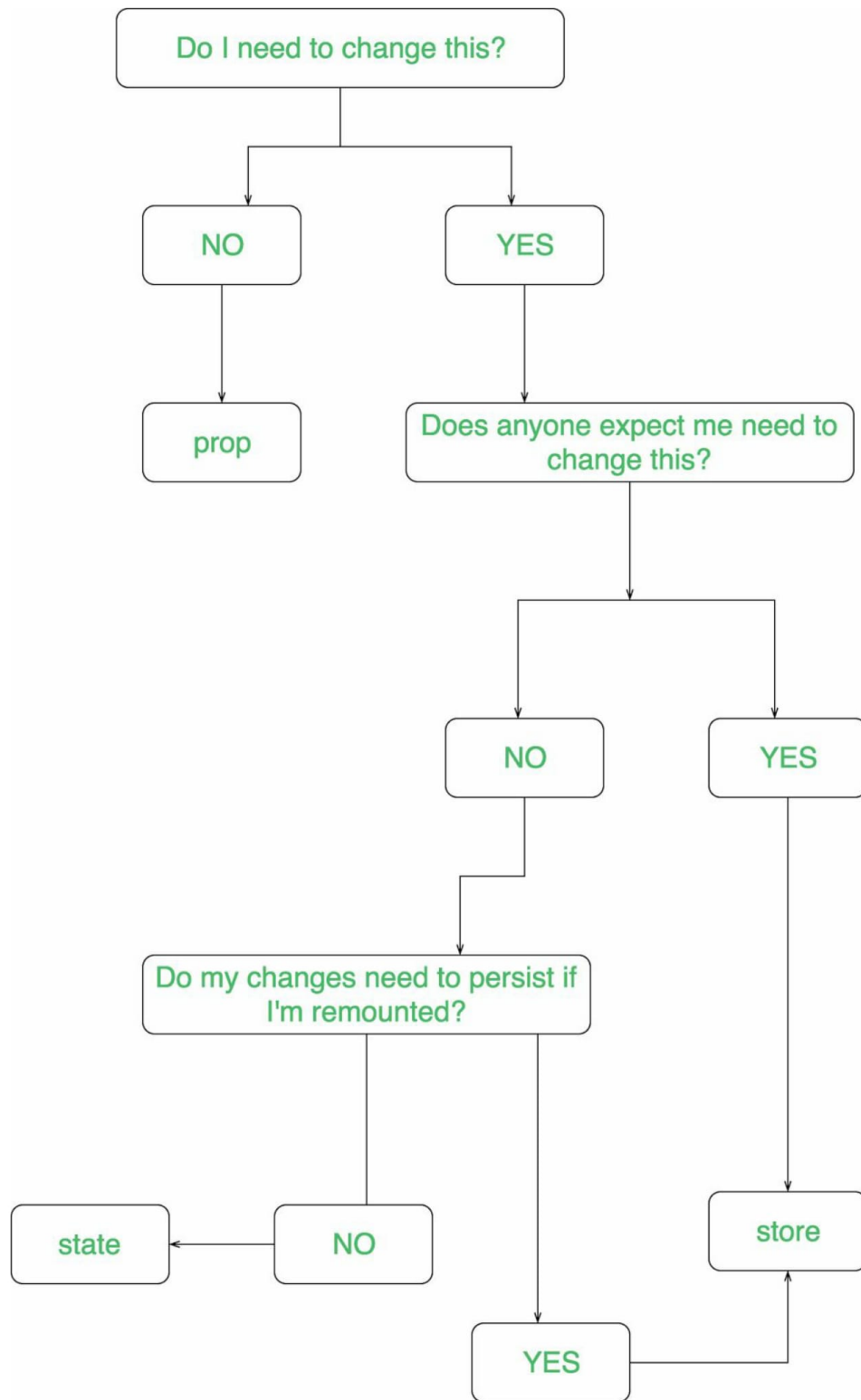
| IN PREVIOUS LECTURE (QUICK RECAP) Date-16/11/2020 | In Today's Lecture (Overview) |
|--|---|
| React Events Adding Events React: Event Handlers Bind this Why Arrow Functions? Passing Arguments Questions For the Self-Practice | React State Creating the state Object Example: Using the state Object React Props React Props Example Question For Self-Practice |

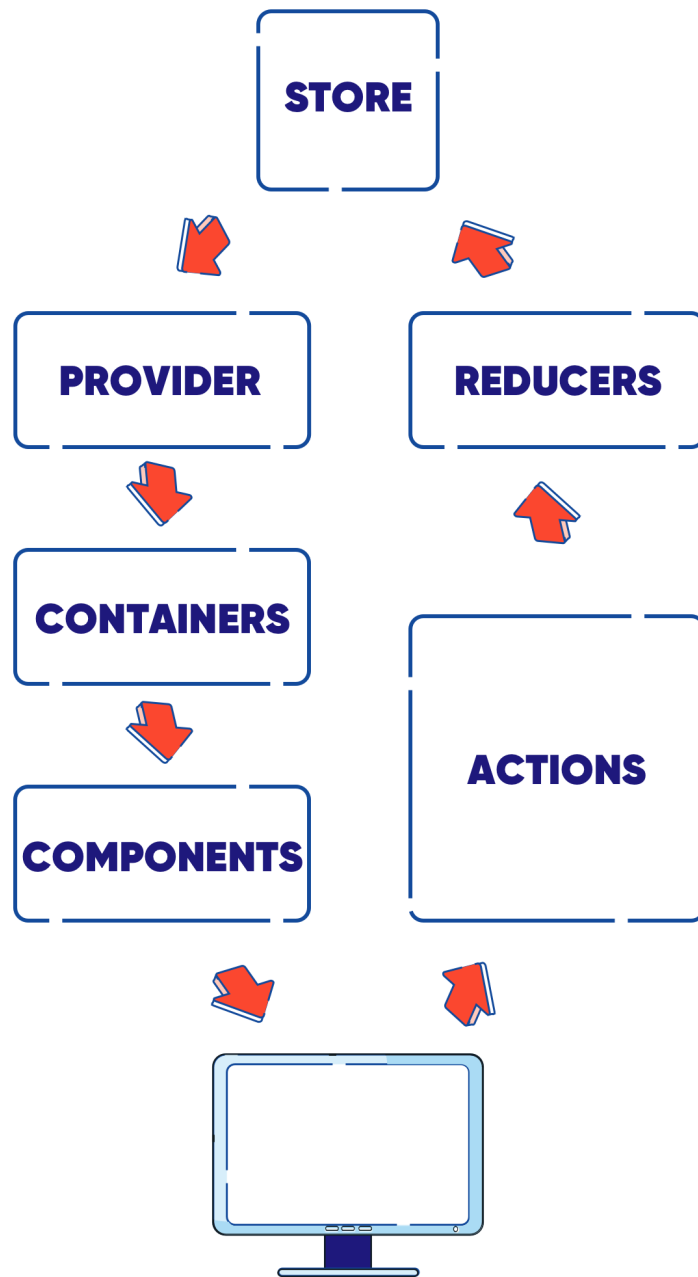
React State

React components has a built-in `state` object.

The `state` object is where you store property values that belongs to the component.

When the `state` object changes, the component re-renders.





Creating the `state` Object

The `state` object is initialized in the constructor:

Example:

Specify the `state` object in the constructor method:

```
class Car extends React.Component {  
  constructor(props) {  
    super(props);  
    this.state = {brand: "Ford"};  
  }  
  render() {  
    return (  
      <div>  
        <h1>My Car</h1>  
      </div>  
    );  
  }  
}
```

The `state` object can contain as many properties as you like:

Example:

Specify all the properties your component need:

```
class Car extends React.Component {  
  constructor(props) {  
    super(props);  
    this.state = {  
      brand: "Ford",  
      model: "Mustang",  
      color: "red",  
      year: 1964  
    };  
  }  
  render() {
```

```
    return (  
      <div>  
        <h1>My Car</h1>  
      </div>  
    );  
  }  
}
```

Using the `state` Object

Refer to the `state` object anywhere in the component by using the `this.state.propertyname` syntax

Example:

Refer to the `state` object in the `render()` method:

```
class Car extends React.Component {  
  constructor(props) {  
    super(props);  
    this.state = {  
      brand: "Ford",  
      model: "Mustang",  
      color: "red",  
      year: 1964  
    };  
  }  
  render() {  
    return ( <  
      div >  
        <  
          h1 > My { this.state.brand } < /h1> <  
          p >  
            It is a { this.state.color } { this.state.model }  
            from { this.state.year }. <  
          /p> <  
        /div>  
      >  
    );  
  }  
}
```

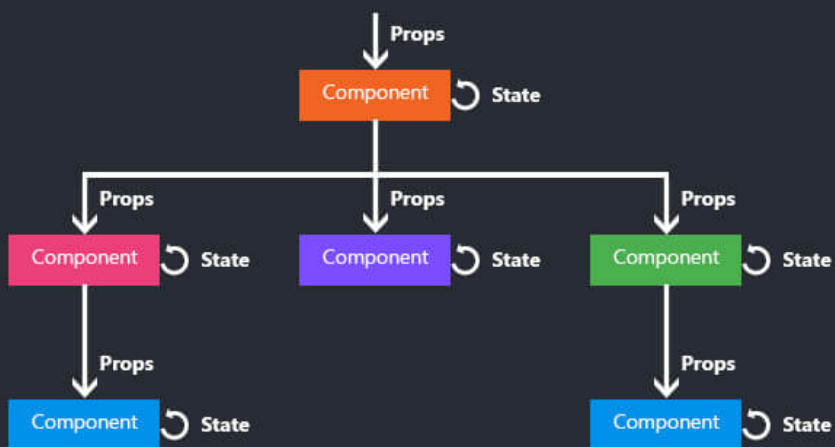
```
}  
  }  
}
```

React Props

Props are arguments passed into React components.

Props are passed to components via HTML attributes.

ReactJS Component State



React Props

React Props are like function arguments in JavaScript *and* attributes in HTML.

To send props into a component, use the same syntax as HTML attributes:

Example

Add a "brand" attribute to the Car element:

```
const myelement = <Car brand="Ford" />;
```

The component receives the argument as a `props` object:

Example

Use the brand attribute in the component:

```
class Car extends React.Component {  
  
  render() {  
  
    return <h2>I am a {this.props.brand}!</h2>;  
  
  }  
  
}
```



Props vs State



- ✓ props are read-only
- ✓ props can not be modified
- ✓ state changes can be asynchronous
- ✓ state can be modified using `this.setState`

State

`this.state.name`

State are mutable

You can define states in the component itself

The state is set and updated by the object.

Both are accessible as attributes of the component class and compose components with a different representation (view)

Props

`this.props.name`

Props are immutable

You can pass properties from parent components

determine the view upon creation, and then they remain static

Both are accessible as attributes of the component class and compose components with a different representation (view)

Question For Self-Practice

https://au-assignment.s3.ap-south-1.amazonaws.com/Week_21_Day_2_Challenge-52e171df-7dad-40b2-8eef-74a28ddbc592.pdf